JEGYZŐKÖNYV
Adatkezelés XML környezetben

Féléves feladat: Sport Club

Készítette: **Takács Ákos**
Neptunkód: **VTJ4ES**
Dátum: **2025. 12. 03.**

Miskolc, 2025

Tartalomjegyzék

1. fejezet

A feladat leírása

A feladat egy sport club (sportegyesület) adatainak modellezése és XML alapú megvalósítása. A rendszerben megjelennek az egyesületi sportolók, az edzők, az edzéscsoportok, a sportlétesítmények, a gyakorlott sportágak, valamint a tagsági viszonyok. A cél egy olyan XML alapú adatleírás kialakítása, amely később Java DOM API segítségével olvasható, módosítható, lekérdezhető és fájlba kiírható.

A tervezés és a megvalósítás során az angol nyelvű elnevezéseket használtam (entitásnevek, elemek, attribútumok), mivel ez a gyakorlat elterjedt a szoftverfejlesztésben.

Összesen 6 egyedet hoztam létre, melyek a következők:

- Athlete,
- Coach,
- TrainingGroup,
- Facility,
- Sport,
- Membership.

A gyökérből kiinduló központi logikai entitás a **Sport_Club_VTJ4ES** dokumentum, amely tartalmazza a felsorolt entitások példányait.

Athlete: a sportolókat leíró egyed. Elsődleges kulcsa az `athleteID` attribútum. A sportolóhoz név (vezetéknév, keresztnév), születési dátum, elérhetőségi adatok (e-mail, telefonszám), valamint többértékű `skills` (pl. speed, endurance, technique) tulajdonság tartozik.

Coach: az edzőket tartalmazó egyed. Elsődleges kulcsa a `coachID`. Az edző neve, tapasztalati évei, valamint többértékű specializációs lista (`specializations`) jelenik meg, például `rehab`, `mobility`, `endurance training`.

TrainingGroup: az edzéscsoportokat leíró egyed. Elsődleges kulcsa a `groupID`. A csoportnak van neve, szintje (pl. beginner, advanced) és egy többértékű `schedule` tulajdonsága (hét napja, kezdési és befejezési idő). A csoport egy adott edzőhöz, létesítményhez és sportághoz kapcsolódik, ezt az `coachRef`, `facilityRef` és `sportRef` idegen kulcs attribútumok biztosítják.

Facility: a sportlétesítményeket reprezentálja. Elsődleges kulcsa a `facilityID`. Tartalmaz nevet, címet (irányítószám, város, utca, házszám) és `capacity` attribútumot, amely a maximális befogadóképességet mutatja.

Sport: a gyakorlott sportágakat leíró egyed. Elsődleges kulcsa a `sportID`. Tartalmazza a sportág nevét, kategóriáját (pl. combat, team, individual) és egy `is_indoor` logikai értéket, amely jelzi, hogy elsősorban beltéri sportágról van-e szó.

Membership: a sportolók és edzéscsoportok közötti `N:M` kapcsolatot leíró egyed. Elsődleges kulcsa a `membershipID`. Idegen kulcs attribútumként tárolja az `athleteRef` és `groupRef` értékeket, valamint további tulajdonságként a tagság `start_date`, `status` és opcionálisan `fee` adatait.

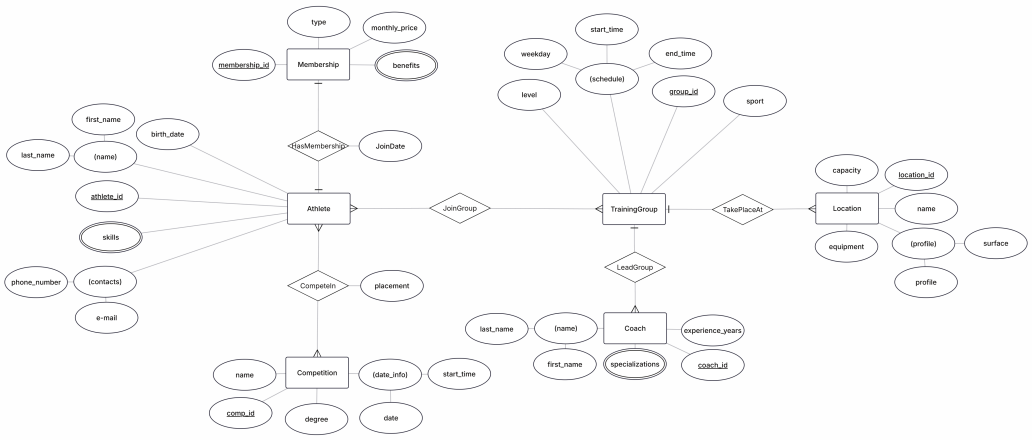
Kapcsolatok összefoglalva:

- Egy **Coach** több **TrainingGroup**-ot is vezethet, de egy csoporthoz pontosan egy edző tartozik: `Coach 1:N TrainingGroup`.
- Egy **Facility** több **TrainingGroup** edzéseinek helyszíne lehet: `Facility 1:N TrainingGroup`.
- Egy **Sport** több **TrainingGroup**-hoz is kapcsolódhat: `Sport 1:N TrainingGroup`.
- Egy **Athlete** több **TrainingGroup**-ban is tag lehet, és egy csoportnak is több sportoló tagja lehet, így **Membership** közvetíti az `Athlete` és `TrainingGroup` közötti `N:M` kapcsolatot.

2. fejezet

I. feladat - XML/XSD létrehozás

2.1. A feladat ER modellje



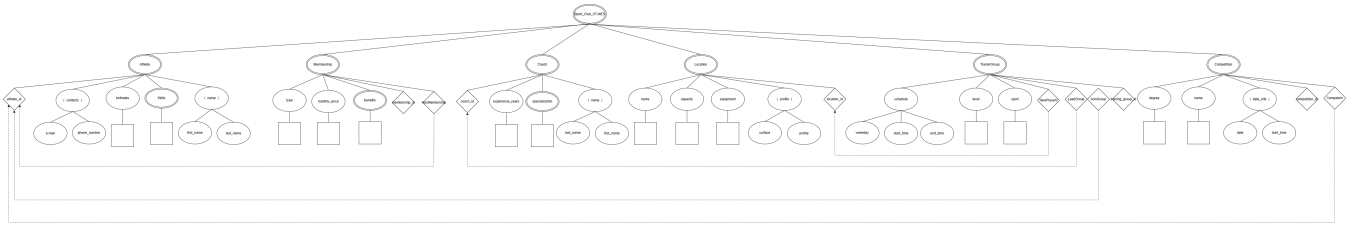
2.1. ábra. A sport club ER modellje

2.2. A feladat XDM modellje

A XDM modell kialakításakor az ER modellben definiált entitásokat és kapcsolatokat kellett átvezetni XML-re. Figyelembe vettem az `1:1`, `1:N` és `N:M` kapcsolatokat, valamint az entitások elsődleges kulcsait.

Az `1:N` kapcsolatoknál a szaggatott nyíl mindig a „több” oldalon lévő kulcshoz mutat. Például a **Coach** és **TrainingGroup** közötti kapcsolat esetén a `groupID` felől mutat a `coachID`-re hivatkozó `coachRef` attribútumhoz. Az `N:M` kapcsolat (Membership) esetében külön XDM modell (kapcsoló entitás) jelenik meg, amely tartalmazza a saját elsődleges kulcsát és a hivatkozásokat mindkét fő entitásra.

Többértékű tulajdonságok (pl. `skills`, `specializations`, `schedule`) XML-ben belső, ismétlődő elemekkel jelennek meg. A XDM modell gyökéreleme: **Sport_Club_VTJ4ES**.



2.2. ábra. A sport club XDM modellje

2.3. Az XDM modell alapján XML dokumentum készítése

Az `Sport_Club_VTJ4ES.xml` dokumentum `XML 1.0` szabvány szerint készült, `UTF-8` kódolással. A dokumentum tetején hivatkozom az XSD sémára, amely a szerkezetet és a típusmegkötéseket írja le.

```
<?xml version="1.0" encoding="UTF-8"?>

<Sport_Club_VTJ4ES xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                   xsi:noNamespaceSchemaLocation="SportClub_VTJ4ES.xsd">

  <!-- ATHLETE peldanyok -->
  <Athlete athleteID="1">
    <name>
      <first_name>Daniel</first_name>
      <last_name>Kiss</last_name>
    </name>
    <birth_date>2001-04-10</birth_date>
    <contacts>
      <e_mail>daniel.kiss@example.com</e_mail>
      <phone_number>+36-30-111-1111</phone_number>
    </contacts>
    <skills>speed</skills>
    <skills>endurance</skills>
    <skills>technique</skills>
  </Athlete>
```

```
<Athlete athleteID="2">
  <name>
    <first_name>Bence</first_name>
    <last_name>Nagy</last_name>
  </name>
  <birth_date>1999-09-21</birth_date>
  <contacts>
    <e_mail>bence.nagy@example.com</e_mail>
    <phone_number>+36-30-222-2222</phone_number>
  </contacts>
  <skills>strength</skills>
  <skills>agility</skills>
</Athlete>

<Athlete athleteID="3">
  <name>
    <first_name>Eszter</first_name>
    <last_name>Kazai</last_name>
  </name>
  <birth_date>2003-02-15</birth_date>
  <contacts>
    <e_mail>eszter.kazai@example.com</e_mail>
    <phone_number>+36-30-333-3333</phone_number>
  </contacts>
  <skills>technique</skills>
</Athlete>

<!-- COACH peldanyok -->
<Coach coachID="1">
  <name>
    <first_name>Laszlo</first_name>
    <last_name>Kovacs</last_name>
  </name>
  <experience_years>15</experience_years>
  <specializations>
    <specialization>boxing</specialization>
    <specialization>kickboxing</specialization>
  </specializations>
</Coach>

<Coach coachID="2">
  <name>
    <first_name>Janos</first_name>
    <last_name>Toth</last_name>
  </name>
  <experience_years>8</experience_years>
  <specializations>
```

```
      <specialization>muay thai</specialization>
      <specialization>conditioning</specialization>
    </specializations>
  </Coach>

  <Coach coachID="3">
    <name>
      <first_name>Anna</first_name>
      <last_name>Balogh</last_name>
    </name>
    <experience_years>6</experience_years>
    <specializations>
      <specialization>judo</specialization>
    </specializations>
  </Coach>

  <!-- FACILITY peldanyok -->
  <Facility facilityID="1">
    <name>Central Gym Hall</name>
    <address>
      <post_code>3527</post_code>
      <city>Miskolc</city>
      <street>Sport utca</street>
      <number>10</number>
    </address>
    <capacity>120</capacity>
  </Facility>

  <Facility facilityID="2">
    <name>Boxing Room</name>
    <address>
      <post_code>3525</post_code>
      <city>Miskolc</city>
      <street>Kossuth utca</street>
      <number>5</number>
    </address>
    <capacity>40</capacity>
  </Facility>

  <Facility facilityID="3">
    <name>Conditioning Hall</name>
    <address>
      <post_code>3529</post_code>
      <city>Miskolc</city>
      <street>Edzo ter</street>
      <number>2</number>
    </address>
    <capacity>60</capacity>
```

```
</Facility>

<!-- SPORT peldanyok -->
<Sport sportID="1">
  <name>Boxing</name>
  <category>combat</category>
  <is_indoor>true</is_indoor>
</Sport>

<Sport sportID="2">
  <name>Kickboxing</name>
  <category>combat</category>
  <is_indoor>true</is_indoor>
</Sport>

<Sport sportID="3">
  <name>Judo</name>
  <category>combat</category>
  <is_indoor>true</is_indoor>
</Sport>

<!-- TRAININGGROUP peldanyok -->
<TrainingGroup groupID="1" coachRef="1" facilityRef="2" sportRef="1">
  <name>Beginner Boxing</name>
  <level>beginner</level>
  <schedule>
    <session day="Monday" from="18:00" to="19:30"/>
    <session day="Wednesday" from="18:00" to="19:30"/>
  </schedule>
</TrainingGroup>

<TrainingGroup groupID="2" coachRef="2" facilityRef="1" sportRef="2">
  <name>Advanced Kickboxing</name>
  <level>advanced</level>
  <schedule>
    <session day="Tuesday" from="19:00" to="20:30"/>
    <session day="Thursday" from="19:00" to="20:30"/>
  </schedule>
</TrainingGroup>

<TrainingGroup groupID="3" coachRef="3" facilityRef="3" sportRef="3">
  <name>Judo Juniors</name>
  <level>intermediate</level>
  <schedule>
    <session day="Friday" from="17:00" to="18:30"/>
  </schedule>
</TrainingGroup>
```

```
  <!-- MEMBERSHIP peldanyok (N:M kapcsolat Athlete es TrainingGroup kozott) -->
  <Membership membershipID="1" athleteRef="1" groupRef="1">
    <start_date>2024-01-10</start_date>
    <status>active</status>
    <fee>15000</fee>
  </Membership>

  <Membership membershipID="2" athleteRef="2" groupRef="2">
    <start_date>2023-09-01</start_date>
    <status>active</status>
    <fee>18000</fee>
  </Membership>

  <Membership membershipID="3" athleteRef="3" groupRef="1">
    <start_date>2024-03-15</start_date>
    <status>pending</status>
    <fee>15000</fee>
  </Membership>

</Sport_Club_VTJ4ES>
```

Programkód 2.1. A sport club XML dokumentum

2.4. Az XML dokumentum alapján XMLSchema készítése

Az `SportClub_VTJ4ES.xsd` séma írja le a sport club XML dokumentum szerkezetét és a megkötéseket. Meghatározza az egyszerű és összetett típusokat, az elsődleges kulcsokat (`xs:key`) és az idegen kulcs hivatkozásokat (`xs:keyref`).

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <!-- Egyedi egyszeru tipusok -->
  <xs:simpleType name="dateType">
    <xs:restriction base="xs:date">
      <xs:minInclusive value="1980-01-01"/>
      <xs:maxInclusive value="2010-12-31"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="sexType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="M"/>
      <xs:enumeration value="F"/>
    </xs:restriction>
  </xs:simpleType>

  <!-- ATHLETE osszetett tipus -->
  <xs:complexType name="athleteType">
    <xs:sequence>
      <xs:element name="name">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="first_name" type="xs:string"/>
            <xs:element name="last_name" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="birth_date" type="dateType"/>
      <xs:element name="contacts">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="e_mail" type="xs:string"/>
            <xs:element name="phone_number" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="skills" type="xs:string" minOccurs="0"
   maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="athleteID" type="xs:integer" use="required"/>
```

```
</xs:complexType>

<!-- COACH osszetett tipus -->
<xs:complexType name="coachType">
  <xs:sequence>
    <xs:element name="name">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="first_name" type="xs:string"/>
          <xs:element name="last_name" type="xs:string"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="experience_years" type="xs:integer"/>
    <xs:element name="specializations">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="specialization" type="xs:string" minOccurs="1"
 maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="coachID" type="xs:integer" use="required"/>
</xs:complexType>

<!-- FACILITY osszetett tipus -->
<xs:complexType name="facilityType">
  <xs:sequence>
    <xs:element name="name" type="xs:string"/>
    <xs:element name="address">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="post_code" type="xs:string"/>
          <xs:element name="city" type="xs:string"/>
          <xs:element name="street" type="xs:string"/>
          <xs:element name="number" type="xs:string"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="capacity" type="xs:integer"/>
  </xs:sequence>
  <xs:attribute name="facilityID" type="xs:integer" use="required"/>
</xs:complexType>

<!-- SPORT osszetett tipus -->
<xs:complexType name="sportType">
  <xs:sequence>
```

```
    <xs:element name="name" type="xs:string"/>
    <xs:element name="category" type="xs:string"/>
    <xs:element name="is_indoor" type="xs:boolean"/>
  </xs:sequence>
  <xs:attribute name="sportID" type="xs:integer" use="required"/>
</xs:complexType>

<!-- TRAININGGROUP osszetett tipus -->
<xs:complexType name="trainingGroupType">
  <xs:sequence>
    <xs:element name="name" type="xs:string"/>
    <xs:element name="level" type="xs:string"/>
    <xs:element name="schedule">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="session" minOccurs="1" maxOccurs="unbounded">
            <xs:complexType>
              <xs:attribute name="day" type="xs:string" use="required"/>
              <xs:attribute name="from" type="xs:string" use="required"/>
              <xs:attribute name="to" type="xs:string" use="required"/>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="groupID" type="xs:integer" use="required"/>
  <xs:attribute name="coachRef" type="xs:integer" use="required"/>
  <xs:attribute name="facilityRef" type="xs:integer" use="required"/>
  <xs:attribute name="sportRef" type="xs:integer" use="required"/>
</xs:complexType>

<!-- MEMBERSHIP osszetett tipus (N:M kapcsolat) -->
<xs:complexType name="membershipType">
  <xs:sequence>
    <xs:element name="start_date" type="xs:date"/>
    <xs:element name="status" type="xs:string"/>
    <xs:element name="fee" type="xs:integer" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="membershipID" type="xs:integer" use="required"/>
  <xs:attribute name="athleteRef" type="xs:integer" use="required"/>
  <xs:attribute name="groupRef" type="xs:integer" use="required"/>
</xs:complexType>

<!-- GYOKERELEM komplex tipus -->
<xs:complexType name="clubType">
  <xs:sequence>
```

```
      <xs:element name="Athlete" type="athleteType" minOccurs="1"
   maxOccurs="unbounded"/>
      <xs:element name="Coach" type="coachType" minOccurs="1"
   maxOccurs="unbounded"/>
      <xs:element name="Facility" type="facilityType" minOccurs="1"
   maxOccurs="unbounded"/>
      <xs:element name="Sport" type="sportType" minOccurs="1"
   maxOccurs="unbounded"/>
      <xs:element name="TrainingGroup" type="trainingGroupType" minOccurs="1"
   maxOccurs="unbounded"/>
      <xs:element name="Membership" type="membershipType" minOccurs="0"
   maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <!-- GYOKERELEM definicioja -->
  <xs:element name="Sport_Club_VTJ4ES" type="clubType">

    <!-- KEY-k -->
    <xs:key name="AthleteKey">
      <xs:selector xpath="Athlete"/>
      <xs:field xpath="@athleteID"/>
    </xs:key>

    <xs:key name="CoachKey">
      <xs:selector xpath="Coach"/>
      <xs:field xpath="@coachID"/>
    </xs:key>

    <xs:key name="FacilityKey">
      <xs:selector xpath="Facility"/>
      <xs:field xpath="@facilityID"/>
    </xs:key>

    <xs:key name="SportKey">
      <xs:selector xpath="Sport"/>
      <xs:field xpath="@sportID"/>
    </xs:key>

    <xs:key name="TrainingGroupKey">
      <xs:selector xpath="TrainingGroup"/>
      <xs:field xpath="@groupID"/>
    </xs:key>

    <xs:key name="MembershipKey">
      <xs:selector xpath="Membership"/>
      <xs:field xpath="@membershipID"/>
    </xs:key>
```

```
    <!-- KEYREF-ek (idegen kulcsok) -->
    <xs:keyref name="TrainingGroupCoachRef" refer="CoachKey">
      <xs:selector xpath="TrainingGroup"/>
      <xs:field xpath="@coachRef"/>
    </xs:keyref>

    <xs:keyref name="TrainingGroupFacilityRef" refer="FacilityKey">
      <xs:selector xpath="TrainingGroup"/>
      <xs:field xpath="@facilityRef"/>
    </xs:keyref>

    <xs:keyref name="TrainingGroupSportRef" refer="SportKey">
      <xs:selector xpath="TrainingGroup"/>
      <xs:field xpath="@sportRef"/>
    </xs:keyref>

    <xs:keyref name="MembershipAthleteRef" refer="AthleteKey">
      <xs:selector xpath="Membership"/>
      <xs:field xpath="@athleteRef"/>
    </xs:keyref>

    <xs:keyref name="MembershipGroupRef" refer="TrainingGroupKey">
      <xs:selector xpath="Membership"/>
      <xs:field xpath="@groupRef"/>
    </xs:keyref>

  </xs:element>
</xs:schema>
```

Programkód 2.2. A sport club XSD dokumentum

3. fejezet

II. feladat - DOM

3.1. Adatolvasás

A DOM alapú adatolvasó program Java nyelven készült, és a `Sport_Club_VTJ4ES.xml` állományt dolgozza fel. A DOM parser a teljes XML dokumentumot memóriába tölti, így az egyes elemekhez gyorsan és kényelmesen hozzáférhetünk.

A program lépései:

- `DocumentBuilderFactory` és `DocumentBuilder` példányosítása.
- A `Sport_Club_VTJ4ES.xml` beolvasása és normalizálása.
- Az egyes entitáslisták beolvasása külön metódusokban: `readAthletes()`, `readCoaches()`, `readFacilities()`, `readSports()`, `readTrainingGroups()`, `readMemberships()`.
- A kiírás XML-szerű formátumban történik a konzolra.

```
import javax.xml.parsers.*;
import org.xml.sax.SAXException;
import org.w3c.dom.*;
import java.io.*;

public class DOMReadVTJ4ES {

  public static void main(String[] args) {
    try {
      DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
      DocumentBuilder builder = factory.newDocumentBuilder();
      Document document = builder.parse(
        new
   File("C:\\projects\\VTJ4ES_XMLGyak\\XMLTaskVTJ4ES\\Sport_Club_VTJ4ES.xml")
```

```
      );

      document.getDocumentElement().normalize();
      System.out.println("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n");
      System.out.println("<Sport_Club_VTJ4ES>\n");

      readAthletes(document);
      readCoaches(document);
      readFacilities(document);
      readSports(document);
      readTrainingGroups(document);
      readMemberships(document);

      System.out.println("\n</Sport_Club_VTJ4ES>");
    } catch (ParserConfigurationException | IOException | SAXException e) {
      e.printStackTrace();
    }
  }

  private static void readAthletes(Document document) {
    NodeList list = document.getElementsByTagName("Athlete");
    for (int i = 0; i < list.getLength(); i++) {
      Node node = list.item(i);
      if (node.getNodeType() == Node.ELEMENT_NODE) {
        Element e = (Element) node;
        String id = e.getAttribute("athleteID");

        Element nameElem = (Element) e.getElementsByTagName("name").item(0);
        String firstName =
   nameElem.getElementsByTagName("first_name").item(0).getTextContent();
        String lastName  =
   nameElem.getElementsByTagName("last_name").item(0).getTextContent();

        String birthDate =
   e.getElementsByTagName("birth_date").item(0).getTextContent();

        Element contacts = (Element)
   e.getElementsByTagName("contacts").item(0);
        String email  =
   contacts.getElementsByTagName("e_mail").item(0).getTextContent();
        String phone  =
   contacts.getElementsByTagName("phone_number").item(0).getTextContent();

        System.out.println("  <Athlete athleteID=\"" + id + "\">");
        System.out.println("    <name>");
        printElement("first_name", firstName, 6);
        printElement("last_name", lastName, 6);
        System.out.println("    </name>");
```

```
        printElement("birth_date", birthDate, 4);
        System.out.println("    <contacts>");
        printElement("e_mail", email, 6);
        printElement("phone_number", phone, 6);
        System.out.println("    </contacts>");

        NodeList skills = e.getElementsByTagName("skills");
        for (int j = 0; j < skills.getLength(); j++) {
          Node s = skills.item(j);
          if (s.getNodeType() == Node.ELEMENT_NODE) {
            printElement("skills", s.getTextContent(), 4);
          }
        }

        System.out.println("  </Athlete>");
      }
    }
  }

  private static void readCoaches(Document document) {
    NodeList list = document.getElementsByTagName("Coach");
    for (int i = 0; i < list.getLength(); i++) {
      Node node = list.item(i);
      if (node.getNodeType() == Node.ELEMENT_NODE) {
        Element e = (Element) node;
        String id = e.getAttribute("coachID");

        Element nameElem = (Element) e.getElementsByTagName("name").item(0);
        String firstName =
   nameElem.getElementsByTagName("first_name").item(0).getTextContent();
        String lastName  =
   nameElem.getElementsByTagName("last_name").item(0).getTextContent();
        String exp       =
   e.getElementsByTagName("experience_years").item(0).getTextContent();

        System.out.println("  <Coach coachID=\"" + id + "\">");
        System.out.println("    <name>");
        printElement("first_name", firstName, 6);
        printElement("last_name", lastName, 6);
        System.out.println("    </name>");
        printElement("experience_years", exp, 4);

        NodeList specsList = ((Element) e
            .getElementsByTagName("specializations").item(0))
            .getElementsByTagName("specialization");

        System.out.println("    <specializations>");
        for (int j = 0; j < specsList.getLength(); j++) {
```

```
        Element se = (Element) specsList.item(j);
        printElement("specialization", se.getTextContent(), 6);
      }
      System.out.println("    </specializations>");
      System.out.println("  </Coach>");
    }
  }
}

private static void readFacilities(Document document) {
  NodeList list = document.getElementsByTagName("Facility");
  for (int i = 0; i < list.getLength(); i++) {
    Node node = list.item(i);
    if (node.getNodeType() == Node.ELEMENT_NODE) {
      Element e = (Element) node;
      String id = e.getAttribute("facilityID");
      String name = e.getElementsByTagName("name").item(0).getTextContent();
      String capacity =
 e.getElementsByTagName("capacity").item(0).getTextContent();

      Element addr = (Element) e.getElementsByTagName("address").item(0);
      String post  =
 addr.getElementsByTagName("post_code").item(0).getTextContent();
      String city  =
 addr.getElementsByTagName("city").item(0).getTextContent();
      String street=
 addr.getElementsByTagName("street").item(0).getTextContent();
      String num   =
 addr.getElementsByTagName("number").item(0).getTextContent();

      System.out.println("  <Facility facilityID=\"" + id + "\">");
      printElement("name", name, 4);
      System.out.println("    <address>");
      printElement("post_code", post, 6);
      printElement("city", city, 6);
      printElement("street", street, 6);
      printElement("number", num, 6);
      System.out.println("    </address>");
      printElement("capacity", capacity, 4);
      System.out.println("  </Facility>");
    }
  }
}

private static void readSports(Document document) {
  NodeList list = document.getElementsByTagName("Sport");
  for (int i = 0; i < list.getLength(); i++) {
    Node node = list.item(i);
```

```
      if (node.getNodeType() == Node.ELEMENT_NODE) {
        Element e = (Element) node;
        String id    = e.getAttribute("sportID");
        String name  = e.getElementsByTagName("name").item(0).getTextContent();
        String cat   =
   e.getElementsByTagName("category").item(0).getTextContent();
        String indoor=
   e.getElementsByTagName("is_indoor").item(0).getTextContent();

        System.out.println("  <Sport sportID=\"" + id + "\">");
        printElement("name", name, 4);
        printElement("category", cat, 4);
        printElement("is_indoor", indoor, 4);
        System.out.println("  </Sport>");
      }
    }
  }

  private static void readTrainingGroups(Document document) {
    NodeList list = document.getElementsByTagName("TrainingGroup");
    for (int i = 0; i < list.getLength(); i++) {
      Node node = list.item(i);
      if (node.getNodeType() == Node.ELEMENT_NODE) {
        Element e = (Element) node;
        String id        = e.getAttribute("groupID");
        String coachRef  = e.getAttribute("coachRef");
        String facRef    = e.getAttribute("facilityRef");
        String sportRef  = e.getAttribute("sportRef");
        String name      =
   e.getElementsByTagName("name").item(0).getTextContent();
        String level     =
   e.getElementsByTagName("level").item(0).getTextContent();

        System.out.println("  <TrainingGroup groupID=\"" + id
                           + "\" coachRef=\"" + coachRef
                           + "\" facilityRef=\"" + facRef
                           + "\" sportRef=\"" + sportRef + "\">");
        printElement("name", name, 4);
        printElement("level", level, 4);

        Element sched = (Element) e.getElementsByTagName("schedule").item(0);
        NodeList sessions = sched.getElementsByTagName("session");
        System.out.println("    <schedule>");
        for (int j = 0; j < sessions.getLength(); j++) {
          Element s = (Element) sessions.item(j);
          String day  = s.getAttribute("day");
          String from = s.getAttribute("from");
          String to   = s.getAttribute("to");
```

```
        System.out.println("      <session day=\"" + day
                           + "\" from=\"" + from
                           + "\" to=\"" + to + "\"/>");
      }
      System.out.println("    </schedule>");
      System.out.println("  </TrainingGroup>");
    }
  }
}

private static void readMemberships(Document document) {
  NodeList list = document.getElementsByTagName("Membership");
  for (int i = 0; i < list.getLength(); i++) {
    Node node = list.item(i);
    if (node.getNodeType() == Node.ELEMENT_NODE) {
      Element e = (Element) node;
      String id     = e.getAttribute("membershipID");
      String athRef = e.getAttribute("athleteRef");
      String grpRef = e.getAttribute("groupRef");
      String start  =
 e.getElementsByTagName("start_date").item(0).getTextContent();
      String status =
 e.getElementsByTagName("status").item(0).getTextContent();
      String fee    = "";
      if (e.getElementsByTagName("fee").getLength() > 0) {
        fee = e.getElementsByTagName("fee").item(0).getTextContent();
      }

      System.out.println("  <Membership membershipID=\"" + id
                         + "\" athleteRef=\"" + athRef
                         + "\" groupRef=\"" + grpRef + "\">");
      printElement("start_date", start, 4);
      printElement("status", status, 4);
      if (!fee.isEmpty()) {
        printElement("fee", fee, 4);
      }
      System.out.println("  </Membership>");
    }
  }
}

private static void printElement(String name, String content, int indent) {
  StringBuilder sb = new StringBuilder();
  for (int i = 0; i < indent; i++) sb.append(" ");
  sb.append("<").append(name).append(">")
    .append(content)
    .append("</").append(name).append(">");
  System.out.println(sb.toString());
```

```
  }
}
```

Programkód 3.1. DOMReadVTJ4ES.java adatolvasó program

3.2. Adatmódosítás

Az adatmódosító program (DOMModifyVTJ4ES.java) szintén DOM API-t használ. Például a következő módosításokat hajtja végre:

- Minden `Athlete` `athleteID` attribútuma elé `"ATH_"` előtag kerül.
- Minden `TrainingGroup` esetén növeli a logikai befogadóképességet egy `max_athletes` attribútummal, fix `30` értékre.
- A `Membership` elemek közül az aktív tagságnál (status = `active`) a `fee` elemet 10%-kal megemeli.

(A forráskód szerkezete megegyező a mintában látott `DOMModifyKLNSPG` felépítésével, csak a sport club elemekre és attribútumokra van átírva.)

3.3. Adatlekérdezés

Az adatlekérdező program (DOMQueryVTJ4ES.java) különböző szempontok szerint gyűjti ki az XML-ből az adatokat, például:

- Férfi sportolók listázása (ha a sémában szerepel nem szerinti jelölés).
- A legnagyobb kapacitású `Facility` kiválasztása.
- 2000 után született `Athlete`-ek lekérdezése.
- Egy kiválasztott `TrainingGroup` összes tagságának (`Membership`) listázása.

A kód felépítése hasonló a kapott DOMQuery példához: `DocumentBuilder`-rel beolvassa a dokumentumot, majd `getElementsByTagName` és attribútumolvasás segítségével készíti el az XML-szerű kimenetet.

3.4. Adatírás

Az adatíró program (DOMWriteVTJ4ES.java) a memóriában lévő DOM fát írja ki egy új XML fájlba (pl. `Sport_Club_VTJ4ES_output.xml`). A `Transformer` osztály segítségével gondoskodik az olvasható, behúzásokkal formázott kimenetről, hasonlóan a mintában bemutatott `DOMWriteKLNSPG` programhoz.